
Project - 1 MA517M-Basic Programming Laboratory	Last Date: 14 November 2026
Name	Roll No.: MA26M015

Integers Modulo n using C++ Classes and Operator Overloading

Objective: To design a C++ program that implements integers modulo n using classes, supports arithmetic operations, and demonstrates operator overloading.

An integer modulo n is an element of the set

$$\mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}$$

with addition and multiplication defined modulo n . Operations include addition, subtraction, multiplication, division (if inverse exists), and comparisons.

Problem Description

Design a class `ModInt` to represent integers modulo n . The class should support arithmetic operations, comparison, and computation of multiplicative inverses using operator overloading.

Class Specification

- **Class Name:** `ModInt`
- **Private Data Members:**
 - `int value;`
 - `int mod;`
- **Public Member Functions:**
 - `ModInt(int v=0, int n=1);`
 - `ModInt inverse() const;`
 - `void display() const;`

Operator Overloading

- `operator +()`, `-`, `*` Performs addition, subtraction, and multiplication modulo n
- `operator /()` Performs division using the multiplicative inverse modulo n (if it exists)
- `operator ==(())`, `!=()` Compares two modular integers
- `operator <()`, `>()`, `<=()`, `>=()` Compares two modular integers by their integer representatives
- `operator <<()` Prints the integer modulo n

Tasks

1. Create modular integers $a = 7 \bmod 5$ and $b = 3 \bmod 5$
2. Compute $a + b$, $a - b$, $a * b$, a/b modulo 5
3. Compute multiplicative inverses of a and b modulo 5
4. Compare a and b using the overloaded comparison operators
5. Display results using the overloaded $<<$ operator

Expected Output Example

`a = 7 mod 5 = 2`

`b = 3 mod 5 = 3`

`a + b = 0 mod 5`

`a - b = 4 mod 5`

`a * b = 1 mod 5`

`a / b = 4 mod 5`

`Inverse of a = 3 mod 5`

`Inverse of b = 2 mod 5`

`a == b : False`

`a < b : True`

Project - 2: Number Slide Game Using C++ Classes

Problem Statement

Design and implement a **Number Slide Game** using **C++ classes**. The program should simulate a 4×4 sliding puzzle where the goal is to arrange numbers from 1 to 15 in order, leaving one empty space. The project should utilize object-oriented programming concepts such as classes, objects, encapsulation, and methods for handling game logic.

Project Requirements

1. Create a **Matrix** class to represent the 4×4 game board.
 - (a) Include a method to generate a random matrix of size 4×4 using `srand()` in the range $[1, 16]$.
 - (b) Include a method to check that all entries of the matrix are distinct.
 - (c) Remove the entry with the value 16 to represent the empty space.
2. Create a **Game** class to manage gameplay.
 - (a) Display options for the user: **Play** or **Solution**.

- (b) If the user chooses **Play**, show the navigation commands:
 - A/a for Left
 - S/s for Down
 - D/d for Right
 - W/w for Up
 - Q/q for Quit
 - (c) When the user presses **Q/q**, confirm again before quitting.
 - (d) After each move, display the updated matrix in a nicely formatted way.
3. If the user chooses **Solution**, display the sequence of movements leading to the solved puzzle, and show the final output.
 4. Ensure proper encapsulation of game logic and board operations within the respective classes.

Suggested Class Structure

1. **Matrix** Class:

- Data member: 4×4 integer array representing the board
- Methods: generateMatrix(), checkDistinct(), removeEmpty(), displayMatrix()

2. **Game** Class:

- Data member: **Matrix** object, user choice
- Methods: playGame(), showSolution(), processMove(char move), confirmQuit()

Reference

For more details about the game, visit: <http://www.artbylogic.com/puzzles/numSlider/numberShuffle.htm>